

2017 Smarter Technical Selling Academy

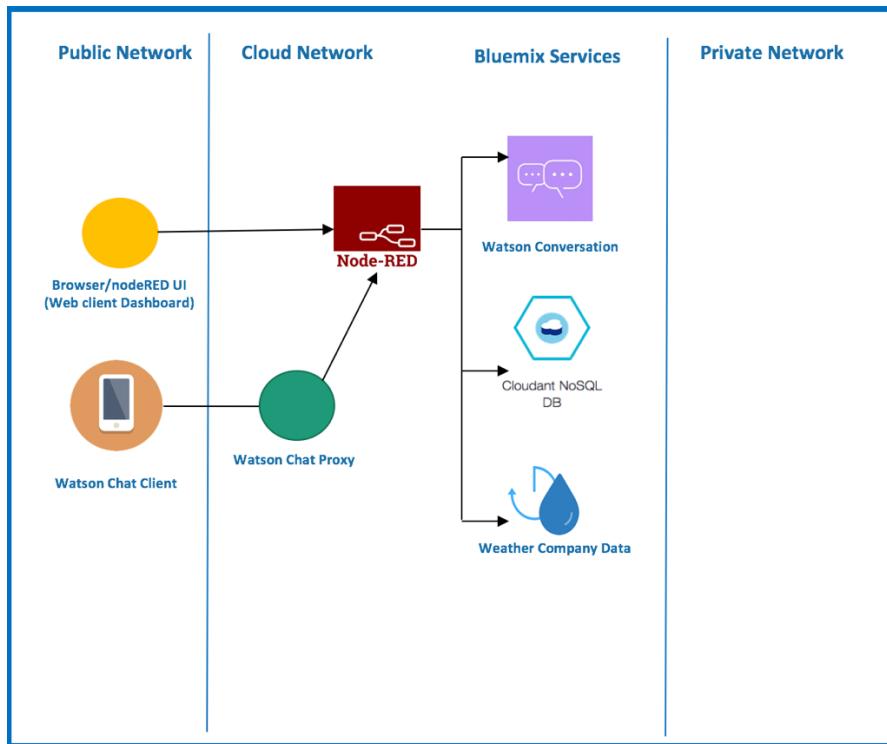
Winning in Act II



Workshop: Watson Conversation Service

Architecture

Below is the architecture overview of the workshop, Watson Conversation Service (WCS). This architecture is consistent with the reference implementations of WCS for cloud native applications using microservices.



Application Overview

This workshop is intended to help you understand the basics of the Watson Conversation Service (WCS) as part of the Watson API's. WCS is a question and answer system that focuses on providing a dialog type of experience between the user and the conversation system. This style of interaction is commonly

Workshop: Watson Conversation Service

called a bot. The intent of this lab is to leverage the WCS capabilities. We will enable through a dialog approach, WCS interacting with data from the weather service API and the ability to issue commands to change the color of our dialog background. Though the example is simple, it will provide you with a solid understanding of the core pieces of WCS.

Terminology

WCS has a couple of terms that need to be understood. This will allow for an easier time with creating an application (i.e. Dialog) in WCS.

Intent: An intent is the intention of the command or question given by the user to WCS. It is common to think of intents as the verbs or actions that need to occur. An Example of an Intent is "Tell me the temperature" or "I want to know the current temperature". In both cases, though the sentences are different they both ask WCS for temperature information. It should be noted that WCS can only support a single Intent per interaction with WCS. So asking questions with multiple intents will produce unreliable ordering of answers to the question.

Entities: An entity is the object that intents use to help narrow the scope of the request. It is common to think of entities as nouns or objects. The nice thing about entities, compared to intents, is that you can have multiple entities per interaction. This is very helpful when trying to narrow down the answers to a question.

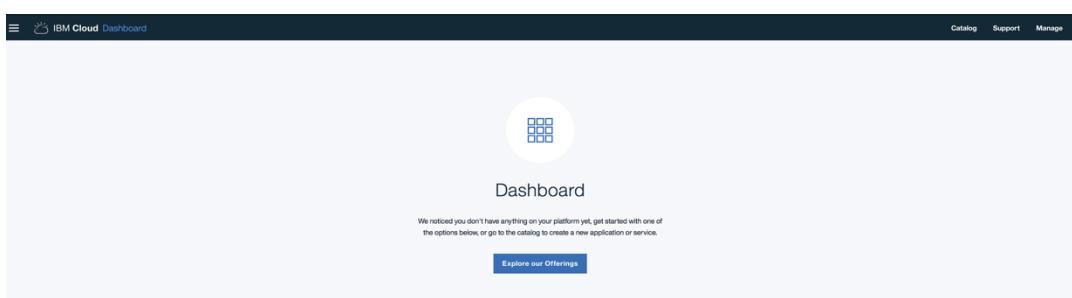
Dialog: The conversation that is created within WCS, is called a dialog. A dialog is composed of creating a flow between intents and entities. The combination of flows and subFlows allows WCS to provide multi-layered conversation based on multiple interactions, instead of a single question and answer.

Prerequisites

You must have a [Bluemix](#) account.

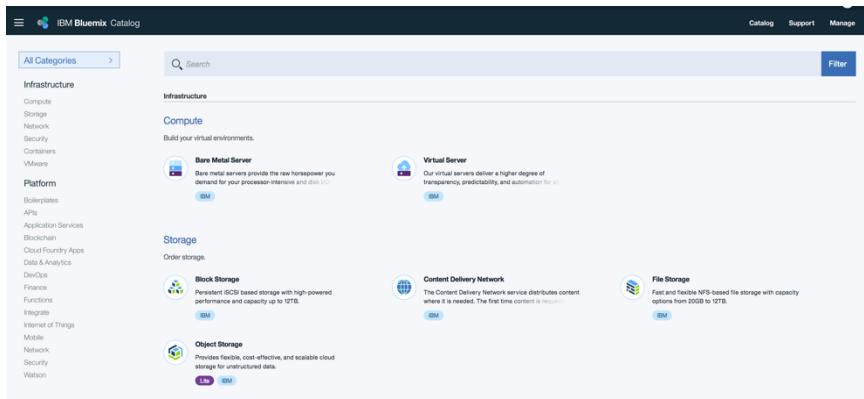
Create Watson Conversation Service

1. Log in to [Bluemix](#)



2. Select the **Catalog** button at the top navigation on the right

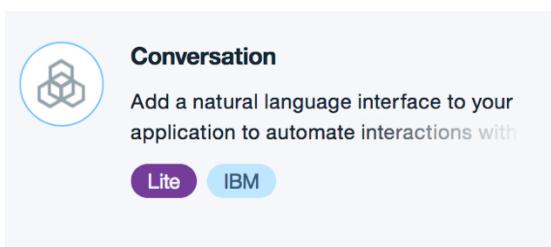
Workshop: Watson Conversation Service



The screenshot shows the IBM Bluemix Catalog interface. On the left, there's a navigation menu with categories like Infrastructure, Platform, and Watson. Under Watson, there are sub-options for Bare Metal Server, Virtual Server, Storage, and Object Storage. A search bar at the top right has the word 'Watson' typed into it. The main content area displays detailed descriptions for each service type.

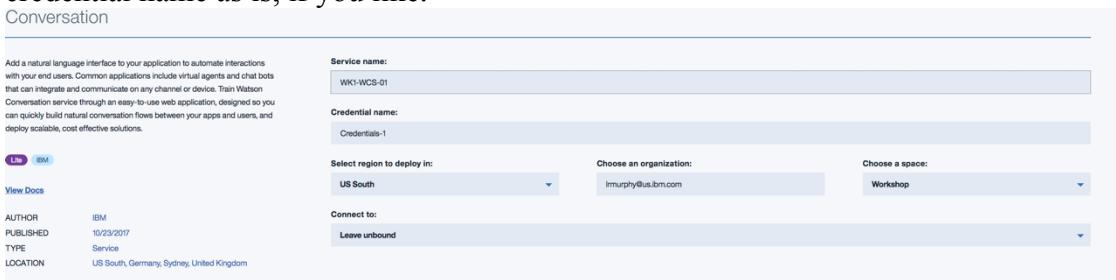
3. Select **Watson** on the left navigation under the Platform menu

4. Select **Conversation**



A card for the 'Conversation' service. It features a circular icon with a hexagonal pattern inside. The title 'Conversation' is at the top, followed by a description: 'Add a natural language interface to your application to automate interactions with'. Below the description are two buttons: 'Lite' (purple) and 'IBM' (blue).

5. In the service name type **WK1-WCS-xx** (where xx is your team number). You can leave credential name as is, if you like.



A screenshot of the 'Create' dialog for the Watson Conversation service. The 'Service name:' field contains 'WK1-WCS-01'. The 'Credential name:' field contains 'Credentials-1'. The 'Select region to deploy in:' dropdown is set to 'US South'. The 'Choose an organization:' dropdown contains 'l Murphy@us.ibm.com'. The 'Choose a space:' dropdown is set to 'Workshop'. At the bottom, there are 'View Docs' and 'Create' buttons. Below the dialog, there's a summary table with columns for AUTHOR, PUBLISHED, TYPE, and LOCATION.

6. Click **Create** in the lower right corner. You should now see the following:



A screenshot of the Watson Conversation service instance details page. The URL is 'Watson / WK1-WCS-01'. The service name is 'WK1-WCS-01'. The main content area is titled 'Conversation' with the same descriptive text as the catalog card. Below the title is a summary of the service's purpose and common applications.

Workshop: Watson Conversation Service

7. You will need to cut and paste your service credentials. Click on **Service Credentials** on the left navigation, right under "Manage". Once the page has loaded, click on **View Credentials** on the right side of the screen. A drop down will show with your specific credentials.

The screenshot shows the 'Service credentials' page for a service named 'WK1-WCS-01'. On the left, there's a navigation bar with 'Manage', 'Service credentials', 'Plan', and 'Connections'. The main area is titled 'Service credentials' and contains a table with one row. The table columns are 'KEY NAME', 'DATE CREATED', and 'ACTIONS'. The single row shows 'Credentials-1' with a creation date of 'Oct 24, 2017 - 02:58:10'. The 'ACTIONS' column contains a copy icon. Below the table, a JSON snippet displays the credential values:

```
{  
  "url": "https://gateway.watsonplatform.net/conversation/api",  
  "username": "0c54cd07-6d0e-45b1-bc59-ff959a66e646",  
  "password": "RgcrJ0Omtk8Z"  
}
```

Your values will be different than shown.

8. Click the **copy icon** to copy the values to your clipboard. Paste the values in a text file for later use.
9. Go back to the Manage page, by clicking on the **Manage** link on the left side navigation. This will take you back to the manage page.

The screenshot shows the 'Conversation' service instance page for 'WK1-WCS-01'. The left sidebar has 'Manage', 'Service credentials', 'Plan', and 'Connections'. The main content area has a purple icon and the title 'Conversation'. It describes adding a natural language interface to an application. A 'Launch tool' button is at the top right. Below it, 'Developer resources' include 'Documentation' and 'Demo' links.

You have now completed creating the WCS service instance.

Launch WCS tooling

1. Click on the **Launch tool** icon to take you to the WCS workspace



Developer resources:

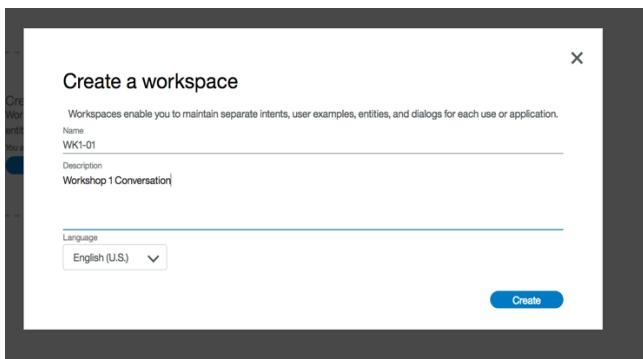
- Documentation
- Demo

Workshop: Watson Conversation Service

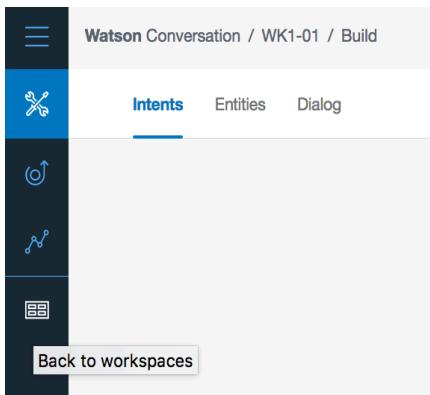
2. This will open another browser window and you should now be on the Watson Conversation Workspace page



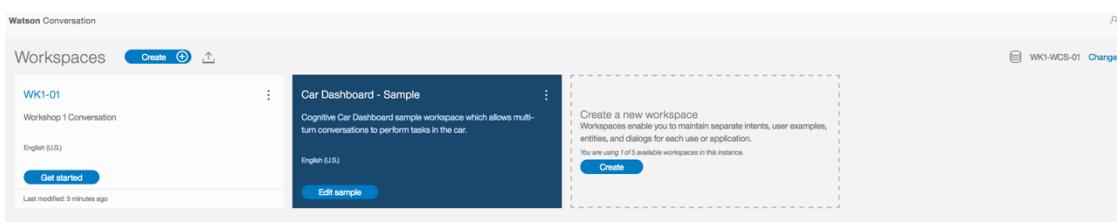
3. You are going to create a new workspace. Click on the **Create a new workspace** tile.
4. Enter a **workspace name** i.e. WK1-xx (where xx is your team number), and select **Create**



5. Click the **last icon (Back to workspaces)** on the left navigation. It should look like

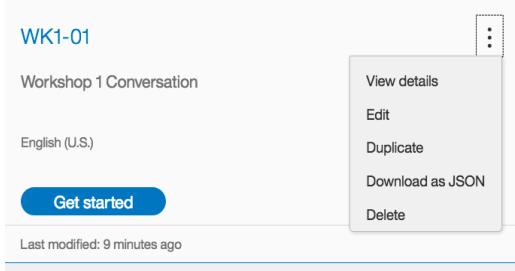


6. You should now see your newly created workspace listed.

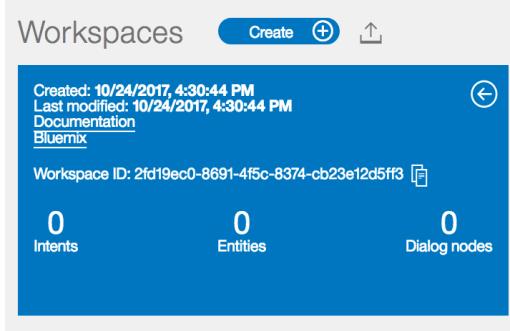


Workshop: Watson Conversation Service

7. Click on the **three grey buttons** in the upper right corner of your workspace tile.

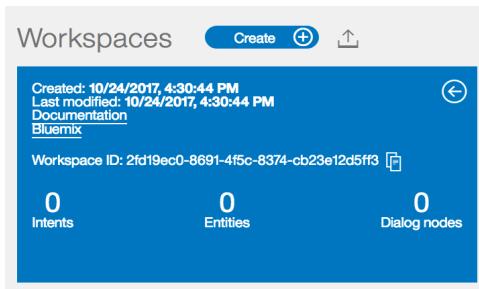


8. Click **View Details**. You should now see something like the following

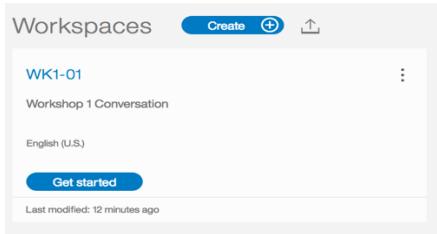


9. Copy the **Workspace ID** and paste it in a text file. **You will need this later in the workshop.**

10. Click on the **White Back Arrow** in the upper corner of the tile.



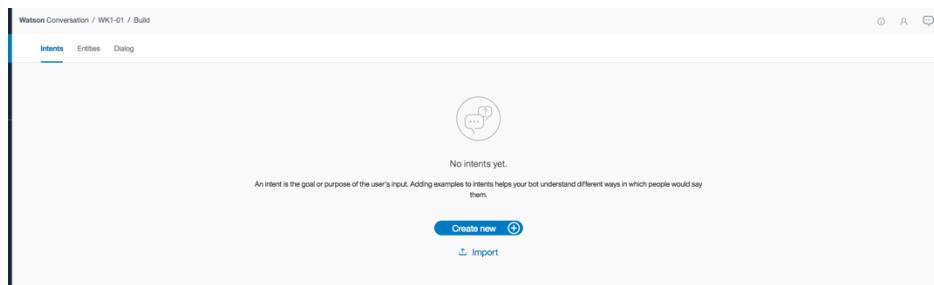
11. Click **Get Started** or anywhere in the tile, to get started.



Create Intents

You are now ready to create Intents! Click on the **Intents** link. You should now see the following:

Workshop: Watson Conversation Service



1. Click the "Create New" button in the middle of the page. You should now see the following:

2. Add a new Intent name of **Information**.

3. You then need to provide some examples. Copy the samples provided, one at a time, and paste under “User example”. Click the “+” button or “Enter” on your keyboard to add each example.

What is the temperature
Can you explain
What is a
I need information about
What is the current temperature
I need some information on the temperature
Can you tell me the temperature
I need to know the temperature

It should look like the following:

Workshop: Watson Conversation Service

Intent name
#Information

User example
Add a user example...

What is the temperature

Can you explain

What is a

I need information about

What is the current temperature

I need some information on the temperature

Can you tell me the temperature

I need to know the temperature

4. Click **Done** in the upper right corner when finished.

Done



5. Add another intent, by clicking **Create new** button.

The screenshot shows the 'Intents' tab selected in the navigation bar. Below it, there are buttons for 'Create new' (with a plus sign), 'Import' (with an upward arrow), 'Export' (with a downward arrow), and 'Delete'. A list of intents is shown, with '#Information' expanded. Under '#Information', the example 'Can you explain' is listed.

6. Use the new Intent name of **Greeting**.

Intents Entities Dialog

Intent name

#Greeting

User example

Add a user example...

Workshop: Watson Conversation Service

7. Copy the samples and click **Done** when finished. Look at the image below.

good evening
Hello
Hi
Howdy
Good morning
Good afternoon
Yo Yo Yo
Sup dude

✓ #Greeting

 Add a new user example...

- Good afternoon
- good evening
- Good morning
- Hello
- Hi
- Howdy
- Sup dude
- Yo Yo Yo

8. Add another intent **GoodBye**. Add the following examples also. Click **Done** when finished.

Good bye
See you later
talk to you later
Goodbye
later dude

✓ #GoodBye

 Add a new user example...

- Goodbye
- Good bye
- later dude
- See you later
- talk to you later

9. Add one last intent **ChangeColor** with the following examples:

I need to change the color of the sensor
Please change the sensor color

Workshop: Watson Conversation Service

Change the color of the sensor
make the sensors color green
change the sensor color to green

▽ #ChangeColor

(+) Add a new user example...

- Change the color of the sensor
- change the sensor color to green
- I need to change the color of the sensor
- make the sensors color green
- Please change the sensor color

10. Click **Done** when finished.

11. You have now finished creating all of the intents. (Nice Job!). You should have created the following intents.

Intents Entities Dialog

Create new (+) Import Export Delete

- > **#ChangeColor**
Change the color of the sensor
- > **#GoodBye**
Good bye
- > **#Greeting**
good evening
- > **#Information**
What is the temperature

Create Entities

Next we want to create some **Entities**. Click on the **Entities** link at the top of the page.

Intents Entities Dialog

My entities System entities

No entities yet.

An entity is a portion of the user's input that you can use to provide a different response to a particular intent. Adding values and synonyms to entities helps your bot learn and understand important details that your users mention.

Create new (+) Use system entities

Import

Workshop: Watson Conversation Service

1. Click the **Create New** button.
2. Type **Temperature** as the new Entity name.

The screenshot shows the 'Entities' tab selected in the top navigation bar. Below it, the word '@Temperature' is typed into the 'Entity' input field. A 'Fuzzy Matching BETA' section is visible, with a note about increasing Watson's ability to recognize misspelled entity values. The main table lists one entity value: 'Add a value, for example, Cat'. To its right, there is a 'Type' dropdown set to 'Value' and a 'Synonyms' dropdown currently set to 'Synonyms'. A link to 'Add synonyms...' is also present.

3. Now notice, you need to add examples of new "Temperature" entities. So add **Current Temperature** as an entity value but you also need to provide some synonyms to help identify variations of the entity value. Now add **Temperature now** and **Current Temp** as synonyms. When you add a synonym, hit the **Enter** key from your keyboard to add the next synonym. After you are finished with each set of synonyms click the **Plus** icon to add them.

The screenshot shows the same interface after adding synonyms. The 'Value' column now includes 'Current Temperature'. In the 'Synonyms' column, three entries are listed: 'Temperature now' and 'Current Temp', with a plus sign (+) icon indicating more can be added. The 'Type' dropdown remains at 'Value'. A 'Done' button and a trash bin icon are visible in the top right corner.

4. You now need to add **Average Temperature** and **Temperature** as entity values with the corresponding synonyms, as shown in the image above. Click **Done** when finished.

Current temperature	current temp	temperature now
Average temperature	Avg temp	Avg temperature
Temperature	Temp	

If for some reason you accidentally don't add one of the synonyms, you can add it after you click the done button. You just select the entity and add the appropriate synonyms.

Workshop: Watson Conversation Service

The screenshot shows the 'Entities' tab selected in the top navigation bar. Under 'My entities', there is a section for '@Temperature'. It includes two buttons: '(+) Add a new synonym value' and '(+) Add a new pattern value'. Below these are three rows of data:

		current temp	temperature now
<input type="checkbox"/>	Current temperature		
<input type="checkbox"/>	Temperature	Temp	
<input type="checkbox"/>	Average temperature	Avg temp	Avg temperature

5. Create another entity called **Degree** also add the associated synonyms. Click **Done** when finished.

Celsius celcius degrees celsius degrees in c degrees c
Fahrenheit degrees fahrenheit degrees f degrees in f

The screenshot shows the '@Degree' entity. It includes two buttons: '(+) Add a new synonym value' and '(+) Add a new pattern value'. Below these are two rows of data:

	celcius	degrees celsius	degrees in c	degrees c
<input type="checkbox"/>	Celsius			
<input type="checkbox"/>	Fahrenheit	degrees fahrenheit	degrees f	degrees in f

6. Add one last entity called **Colors** and the associated values. Click **Done** when finished.

Yellow
Blue
White
Green
Red
Black off

The screenshot shows the '@Colors' entity. It includes two buttons: '(+) Add a new synonym value' and '(+) Add a new pattern value'. Below these are five rows of data, each with an input checkbox and an 'Add synonyms...' button:

<input type="checkbox"/>	Blue	Add synonyms...
<input type="checkbox"/>	White	Add synonyms...
<input type="checkbox"/>	Green	Add synonyms...
<input type="checkbox"/>	Red	Add synonyms...
<input type="checkbox"/>	Black	off

7. Great you now completed the adding of Entities. (Well done!). You should have created the following entities.

Workshop: Watson Conversation Service

The screenshot shows the 'Entities' tab selected in the top navigation bar. Below it, the 'My entities' section lists three entities:

- @Colors: Yellow, White, Green, Blue, Black, Red
- @Degree: Celsius, Fahrenheit
- @Temperature: Average temperature, Temperature, Current temperature

At the bottom of the list are buttons for 'Create new', 'Import', 'Export', and 'Delete'.

Create a Dialog

We are now ready to create a dialog. Click on the **Dialog** link at the top of the page.

The screenshot shows the 'Dialog' tab selected in the top navigation bar. A large circular icon with a question mark and three dots is centered. Below it, the text 'No dialog yet' is displayed. A small note below explains what a dialog is. At the bottom is a 'Create' button with a plus sign.

1. Click the **Create** button to start to create a dialog. You should now see the following:

The screenshot shows the dialog tree. It starts with a root node 'WK1-01' which branches into two nodes:

- Welcome**: welcome
- Anything else**: anything_else

Each node has a '1 Response / 0 Context set' status and three blue circular icons on the right.

What you see are two "Dialog Nodes". The first is the standard "Welcome" message and the other is a catch-all "Anything else". Remember the way a conversation dialog works is by scanning from the top of the tree and evaluating every node until a condition is met that satisfies the question being asked. So

Workshop: Watson Conversation Service

when the conversation starts WCS will respond with the "Welcome" Node. If you click on the "Welcome" node you will see that the standard Watson response is "Hello. How can I help you?"

The screenshot shows the Watson Conversation Service interface. On the left, there's a tree view of nodes under 'WK1-02'. The 'Welcome' node is selected, highlighted in blue. It has one child node, 'Anything else'. On the right, the configuration panel for the 'Welcome' node is displayed. It shows the node name 'Welcome' and a 'Customize' button. Below it, the 'If bot recognizes:' section lists 'welcome' with a minus sign and a plus sign. The 'Then respond with:' section contains a single response: '1. Hello. How can I help you?' with a minus sign. There's also a placeholder 'Add a variation to this response'. At the bottom, it says 'And finally' and 'Wait for user input'.

2. We now want to create our own node, based on the "Intents" we create earlier. To add a new node in the tree, click on the **Welcome** node, and then click on the **Add Node** icon.

This screenshot is identical to the one above, showing the Watson Conversation Service interface. The tree view on the left shows the 'Welcome' node and its child 'Anything else' node. The configuration panel on the right is the same, with the 'Welcome' node settings and a single response '1. Hello. How can I help you?'. The 'Wait for user input' dropdown is also present.

3. You should now see the following:

Workshop: Watson Conversation Service

The screenshot shows the Watson Conversation Service Dialog editor interface. At the top, there are tabs for 'Intents', 'Entities', and 'Dialog'. The 'Dialog' tab is selected, indicated by a blue underline. Below the tabs, there are two buttons: 'Add node' (blue) and 'Add child node' (grey). On the left, a tree view displays a root node 'WK1-02' with three children: 'Welcome', 'No condition set', and 'Anything else'. Each node has a small info icon (three dots) on its right. The 'Welcome' node contains the intent 'welcome'. To the right of the tree, there is a configuration panel for a new node. It includes fields for 'Name this node...' (with a 'Customize' button), 'If bot recognizes:' (with a dropdown placeholder 'Enter an intent, entity or context variable...'), 'Then respond with:' (with a text input field 'Enter a response...'), and 'And finally' (with a dropdown placeholder 'Wait for user input'). A large blue 'X' button is located in the top right corner of the configuration panel.

4. Type **Greetings** as the name of the node. “If bot recognize” should be **#Greeting**. You can provide any response text you like. It should look similar to the following:

The screenshot shows the configuration for the 'Greetings' dialog node. The node name is 'Greetings'. The 'If bot recognizes:' section contains the intent '#Greeting'. The 'Then respond with:' section contains a single response: '1. Hey there, can I help you today?'. Below the responses, there is a field for 'Add a variation to this response'. At the bottom, there is a section for 'And finally' with a dropdown placeholder 'Wait for user input'.

5. You can click the **blue X** in the upper right to close the dialog node editor.
6. Again add another root node by clicking on the **Greetings** node and then clicking **Add node**. This node's values are as follows:

Workshop: Watson Conversation Service

Goodbye

[Customize](#)



If bot recognizes:

#GoodBye [-](#) [+](#)

Then respond with:

1. Have a nice day, it was a pleasure working with you! [-](#)

Add a variation to this response

And finally

Wait for user input [▼](#)

7. You can click the **blue X** in the upper right to close the dialog node editor.
8. Add a new node under the **Goodbye** node, called **Change Color**.

Change Color

[Customize](#)



If bot recognizes:

#ChangeColor [-](#) [+](#)

Then respond with:

Enter a response...

And finally

Wait for user input [▼](#)

9. In this node, your response needs to have a condition set, prior to responding. This is done by clicking on the **Customize** link in the upper right. Change **Multiple responses** to **On**.

Workshop: Watson Conversation Service

Customize "Change Color"

Slots ⓘ off

Enable this to gather the information your bot needs to respond to a user within a single node.

Prompt for everything

Enable this to ask for multiple pieces of information in a single prompt, so your user can provide them all at once and not be prompted for them one at a time.

Multiple responses ⓘ on

Enable multiple responses so that your bot can provide different responses to the same input, based on other conditions.

[Cancel](#) [Apply](#)

10. Click **Apply**. You should now see the following:

Change Color [Customize](#) [X](#)

If bot recognizes:

#ChangeColor [-](#) [+](#)

Then respond with:

If bot recognizes	Respond with
1 Enter an intent, entity or context variab -	Enter a response... ⚙️ trash

[+ Add response](#)

And finally

Wait for user input [▼](#)

11. Make the response condition **@Colors**. For the “respond with” input:

I just changed the color to <? entities.Colors.literal ?>

Workshop: Watson Conversation Service

Change Color

If bot recognizes:

#ChangeColor - +

Then respond with:

If bot recognizes	Respond with
1 @Colors	I just changed the color to <? entities ⚙️ ✖️

+ Add response

Take special note of the extra information in the first response. Make sure to copy it properly.

This is an expression language snippet of code. If you look at the information in the node, the intent is to change color and the first condition for a response is "@Colors", which signals to Watson, if a known color is provided in the user's input then use that color in the response.

12. We are going to add a single line of code to the JSON. The purpose of this code, is to send a signal to the NodeRed application that some special processing needs to occur. So for Watson to respond to the @Color entity request, we need to have a new attribute on the JSON object.
13. To change the JSON, select the **Settings Wheel**, which is located to the right of the “Respond with” entry. You should now see:

Configure response 1

If bot recognizes:

@Colors - +

Then respond with:

1. I just changed the color to <? entities.Colors.literal ?> ✖️

Add a variation to this response

14. Select the **3 blue dots and Open JSON editor**

Workshop: Watson Conversation Service

Configure response 1

If bot recognizes:

@Colors  

Then respond with:

1. I just changed the color to <? entities.Colors.literal ?>

[Open JSON editor](#)

Add a variation to this response

15. You should now see:

Configure response 1

If bot recognizes:

@Colors  

Then respond with:

```
1 {
2   "output": {
3     "text": {
4       "values": [
5         "I just changed the color to <? entities.Colors.literal ?>"
6       ]
7     }
8   }
9 }
```

16. Type "**action": "changeColor"** make sure you put the "**comma**" right after the curly bracket, but before the new code you just added. When done, click **Save**

Workshop: Watson Conversation Service

If bot recognizes:

@Colors (−) (+)

Then respond with:

```
1 { "output": {
2   "text": {
3     "values": [
4       "I just changed the color to <? entities.Colors.literal ?>"
5     ]
6   },
7   "action": "changeColor"
8 }
9 }
10 }
```



17. Select **Add response** to add another response. Set the response condition to **true**, and for the “respond with” input:

I do not know that color, please train me and try again.

18. You should now have the following. When finished, click the **blue X** in the upper right to close the dialog node editor

Change Color @ Customize

If bot recognizes:

#ChangeColor (−) (+)

Then respond with:

If bot recognizes	Respond with
1 @Colors	I just changed the color to <? entities
2 true	I do not know that color, please train

[\(+\) Add response](#)

And finally

Wait for user input (−) (+)

If an unknown color is typed (i.e. Violet) in the user's input, the processing will hit the "true" condition. This is because "Violet" is not in our "@Color" entity definition.

Workshop: Watson Conversation Service

19. Add a node under "Change Color" called **What else**, with the following information. When finished, click the **blue X** in the upper right to close the dialog node editor

What else

If bot recognizes:

true

Then respond with:

1. Is there anything else I can help you with?

Add a variation to this response

And finally

Wait for user input

20. At this point your Dialog Tree should look like the following

Workshop: Watson Conversation Service

The screenshot shows the Watson Conversation Service dialog builder interface. At the top, there are tabs for 'Intents', 'Entities', and 'Dialog'. The 'Dialog' tab is selected, indicated by a blue underline. Below the tabs are two buttons: 'Add node' (highlighted in blue) and 'Add child node'. The main area displays a list of nodes under a header 'WK1-01'. The nodes are listed vertically with their names, intents, and response counts:

- Welcome welcome 1 Response / 0 Context set
- Greetings #Greeting 1 Response / 0 Context set
- Goodbye #Goodbye 1 Response / 0 Context set
- Change Color #ChangeColor 2 Responses / 0 Context set
- What else True 1 Response / 0 Context set
- Anything else anything_else 1 Response / 0 Context set

Each node has a small blue three-dot menu icon on its right side.

21. We are now on the last node we are going to create. Insert this node between the "Goodbye" and "Change Color" node. This is done by clicking on the **Goodbye** node and then clicking **Add Node**.

Workshop: Watson Conversation Service

The screenshot shows the 'Dialog' tab selected in the top navigation bar. Below it, there are two buttons: 'Add node' (highlighted in blue) and 'Add child node'. A list of nodes is displayed under the heading 'WK1-01'. The nodes are:

- Welcome (welcome): 1 Response / 0 Context set
- Greetings (#Greeting): 1 Response / 0 Context set
- Goodbye (#Goodbye): 1 Response / 0 Context set
- No condition set: 0 Responses / 0 Context set
- Change Color (#ChangeColor): 2 Responses / 0 Context set
- What else (True): 1 Response / 0 Context set
- Anything else (anything_else): 1 Response / 0 Context set

22. Enter **Information** as the name of the node. In the "If bot recognizes" type **#Information**. Do not provide any responses.

The screenshot shows the configuration for the 'Information' dialog node. It includes sections for 'If bot recognizes', 'Then respond with', and 'And finally'.

- If bot recognizes:** #information
- Then respond with:** Enter a response...
- And finally:** Wait for user input

23. Click the **blue X** in the upper right to close the dialog node editor

24. Now you want to create a sub-dialog. This time make sure the **Information** node is selected and then click on the **Add child node** button.

Workshop: Watson Conversation Service

The screenshot shows the Watson Conversation Service interface with the 'Dialog' tab selected. A node named 'WK1-01' is expanded, revealing its child nodes:

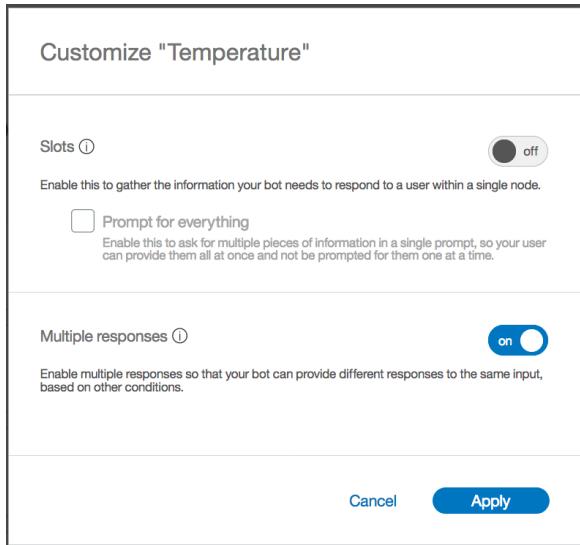
- Welcome (welcome): 1 Response / 0 Context set
- Greetings (#Greeting): 1 Response / 0 Context set
- Goodbye (#Goodbye): 1 Response / 0 Context set
- Information (#Information): 0 Responses / 0 Context set (This node is highlighted with a blue border)
- Change Color (#ChangeColor): 2 Responses / 0 Context set
- What else (True): 1 Response / 0 Context set
- Anything else (anything_else): 1 Response / 0 Context set

25. Enter **Temperature** as the name of the node.
26. In the "If bot recognizes" field enter **@Temperature:(Current Temperature)**. This is the entity defined earlier.

The screenshot shows the configuration for the 'Temperature' node. The 'Name' field is set to 'Temperature'. The 'If bot recognizes:' field contains the entity '@Temperature:(Current temperature)'. The 'Then respond with:' section is currently empty.

27. We want to be able to add multiple responses. To do this, select the **Customize** link in the upper right corner. Turn **On** the option “Multiple Responses.” Select **Apply**

Workshop: Watson Conversation Service



28. You should now see:

The screenshot shows the Watson Conversation Service interface. At the top, it says 'Temperature'. Below it, there's a section for 'If bot recognizes' with the entry '@Temperature:(Current temperature)'. Under 'Then respond with', there's a table with one row. The first column is 'If bot recognizes' with the value 'Enter an intent, entity or context variab'. The second column is 'Respond with' with the value 'Enter a response...'. At the bottom, there's a note 'And finally' followed by 'Wait for user input' with a dropdown arrow.

29. In the Response section, for the first response, type **@Degree:Celsius** in the “if bot recognizes” field. Then add the Watson response of “The current temperature in Celsius is”.

Workshop: Watson Conversation Service

Temperature Customize X

If bot recognizes:

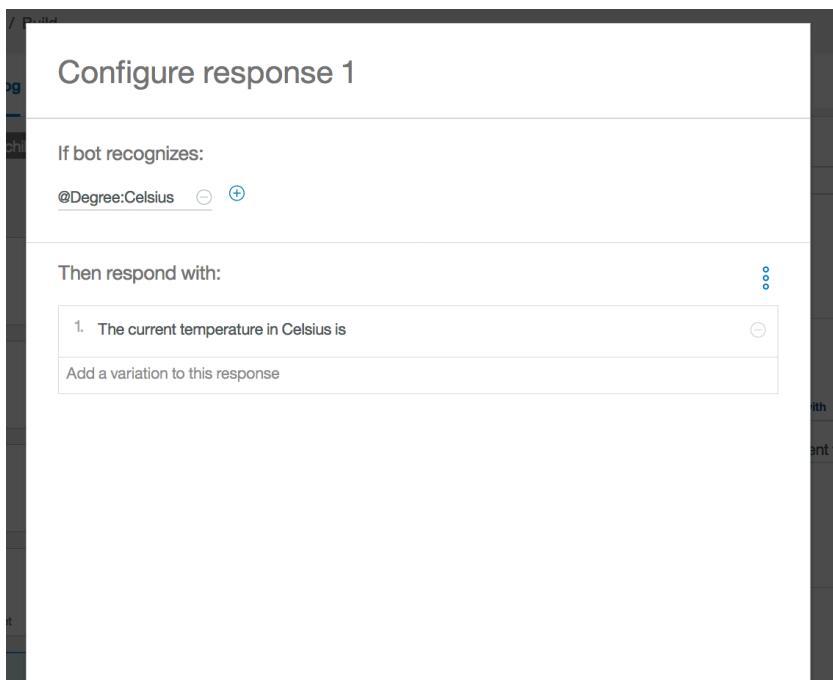
@Temperature:(Current temperature) ⊖ ⊕

Then respond with:

If bot recognizes	Respond with
1 @Degree:Celsius ⊖	The current temperature in Celsius is ⚙️ 🗑️

⊕ Add response

- 30.** We want to add another Watson Response for @Degree:Celsius. To do this, select the **Settings Wheel**, which is located to the right of the “Respond with” entry. You should now see:



Configure response 1

If bot recognizes:

@Degree:Celsius ⊖ ⊕

Then respond with:

1. The current temperature in Celsius is ⊖

Add a variation to this response

- 31.** Within the “Add a variation to this response”, type “**The temperature in Celsius currently is**”. Select “**Save**” when done.

Workshop: Watson Conversation Service

Configure response 1

If bot recognizes:

@Degree:Celsius ⊖ ⊕

Then respond with:

1. The current temperature in Celsius is	⊖
2. The temperature in Celsius currently is	⊖
Add a variation to this response	

Variations are **sequential**. Set to random ⓘ

32. Click **Add response**, to add another response condition of **@Degree:Fahrenheit** with the response message of "**The current temperature in Fahrenheit is**".

Temperature

_customize Customize



If bot recognizes:

@Temperature:(Current temperature) ⊖ ⊕

Then respond with:

If bot recognizes	Respond with
1 @Degree:Celsius	The current temperature in Celsius is ⊖ ⚙️ ⓧ
2 @Degree:Fahrenheit	The current temperature in Fahrenheit ⊖ ⚙️ ⓧ

⊕ Add response

33. Add one last response condition. This is a catch all condition for this node. Set the "if bot recognizes" to **true**. Add the Watson response as "**The current temperature in Fahrenheit is**". The reason for this last response to have a response in the event Watson doesn't know what type of temperature you are looking for. For example, Watson knows you are looking for information

Workshop: Watson Conversation Service

about the temperature, but isn't sure if you mean in celsius or fahrenheit. Your completed node should look the like the following:

Temperature Customize X

If bot recognizes:
@Temperature:(Current temperature) ⊖ ⊕

Then respond with:

If bot recognizes	Respond with
1 @Degree:Celsius	The current temperature in Celsius is ⊖ ⓧ
2 @Degree:Fahrenheit	The current temperature in Fahrenheit ⊖ ⓧ
3 true	The current temperature in Fahrenheit ⊖ ⓧ

⊕ Add response

And finally

Wait for user input ▼

34. We are going to add a single line of code to the JSON. The purpose of this code, is to send a signal to the NodeRed application that some special processing needs to occur. So for Watson to respond to the **degree:Celsius** entity request, we need to have a new attribute on the JSON object.
35. To change the JSON, select the **Settings Wheel** beside the “Respond with” entry for the condition “@Degree:Celsius”. You should see:

Configure response 1

If bot recognizes:
@Degree:Celsius ⊖ ⊕

Then respond with:

1. The current temperature in Celsius is ⊖
2. The temperature in Celsius currently is ⊖

Add a variation to this response
Variations are sequential. Set to random ⊕

Workshop: Watson Conversation Service

36. Select the **3 blue buttons**, and **Open JSON editor**

Configure response 1

If bot recognizes:

@Degree:Celsius  

Then respond with:

1. The current temperature in Celsius is

 Open JSON editor

2. The temperature in Celsius currently is



Add a variation to this response

Variations are **sequential**. Set to random 

37. You should now see:

Configure response 1

If bot recognizes:

@Degree:Celsius  

Then respond with:

```
1 {  
2   "output": {  
3     "text": {  
4       "values": [  
5         "The current temperature in Celsius is",  
6         "The temperature in Celsius currently is"  
7       ],  
8       "selection_policy": "sequential"  
9     }  
10  }  
11 }
```

38. Type **"action": "CurrentTempCelsius"** make sure you put the **"comma"** right after the curly bracket, but before the new code you just added. Select **Save** when done.

Workshop: Watson Conversation Service

```
1 @Degree:Celsius
2
3 {
4     "output": {
5         "text": {
6             "values": [
7                 "The current temperature in Celsius is",
8                 "The temperature in Celsius currently is"
9             ],
10            "selection_policy": "sequential"
11        },
12        "action": "CurrentTempCelsius"
13    }
14 }
```

39. You now need to do the same thing for each of the other responses on the temperature node.
For the **degree:Fahrenheit** response add "action": "CurrentTempFahrenheit"

If bot recognizes:

@Degree:Fahrenheit

Then respond with:

```
1 {
2     "output": {
3         "text": {
4             "values": [
5                 "The current temperature in Fahrenheit is"
6             ],
7             "selection_policy": "sequential"
8         },
9         "action": "CurrentTempFahrenheit"
10    }
11 }
```

40. Now do it for the **true** condition as well. We will have the default response be in fahrenheit so
add "action": "CurrentTempFahrenheit"

Workshop: Watson Conversation Service

If bot recognizes:

true  

Then respond with:

```
1 {
2   "output": {
3     "text": {
4       "values": [
5         "The current temperature in Fahrenheit is"
6       ]
7     },
8     "action": "CurrentTempFahrenheit"
9   }
10 }
```

41. Click the **blue X** in the upper right to close the dialog node editor.

42. We have one last sub-node to create. Under the **Temperature** node, create a new node. This is done by clicking on the **Temperature** node and then clicking on **Add node** button.

Workshop: Watson Conversation Service

The screenshot shows the Watson Conversation Service interface in the 'Dialog' tab. At the top, there are buttons for 'Add node' and 'Add child node'. Below this, a list of nodes is displayed under a parent node named 'WK1-01'. The nodes are:

- Welcome (welcome) - 1 Response / 0 Context set
- Greetings (#Greeting) - 1 Response / 0 Context set
- Goodbye (#Goodbye) - 1 Response / 0 Context set
- Information (#Information) - 0 Responses / 0 Context set
 - Temperature (@Temperature:(Current temperature)) - 3 Responses / 0 Context set
- Change Color (#ChangeColor) - 2 Responses / 0 Context set
- What else (True) - 1 Response / 0 Context set

43. Call it **Unknown**. Add the information like below. When finished, click the **blue X** in the upper right to close the dialog node editor

Workshop: Watson Conversation Service

Unknown Customize X

If bot recognizes:

true () +

Then respond with:

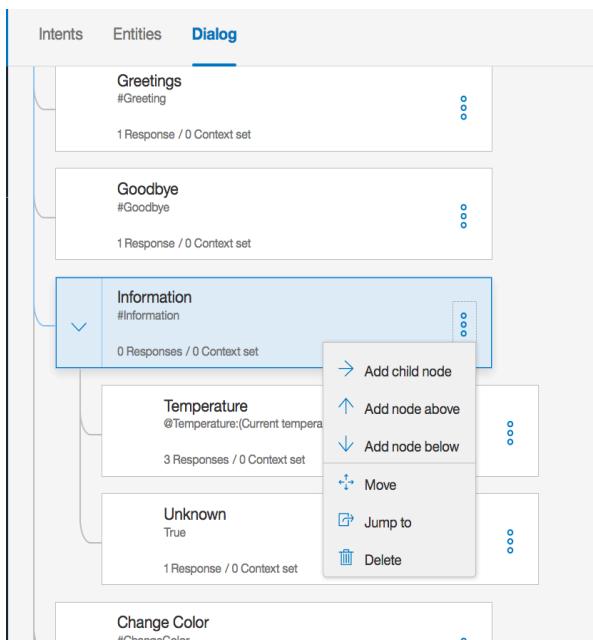
1. Sorry, I don't know what you are asking for, please try again ()

Add a variation to this response

And finally

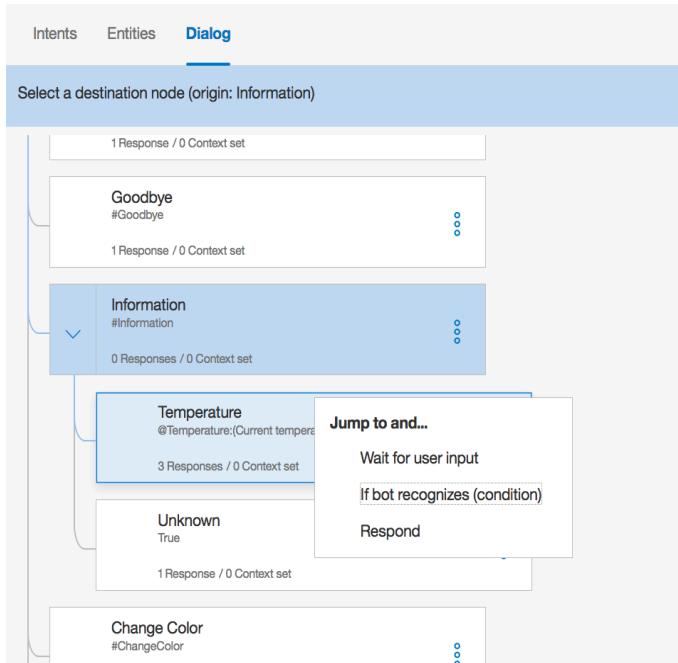
Wait for user input ✓

44. The next step is to go back to the **Information** node, by clicking on the node itself. Next click on the **3 blue buttons** This will open a small menu like the following:

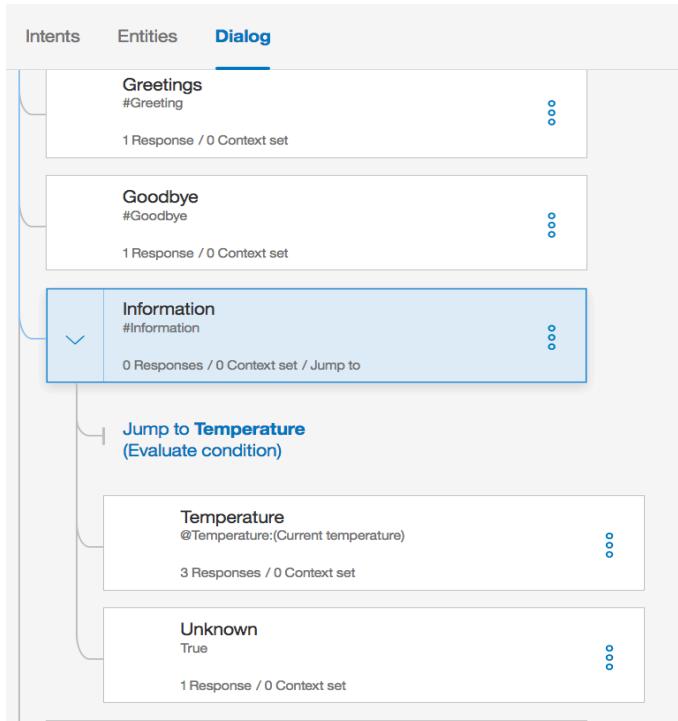


45. Click on the **Jump to** menu item and then click on the **Temperature** node. You should see the following:

Workshop: Watson Conversation Service

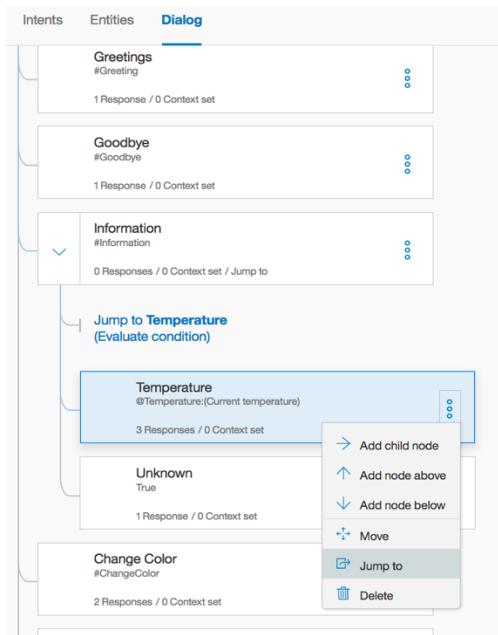


46. Click on the **If Bot recognized condition** menu item. Your screen should now look like the following:

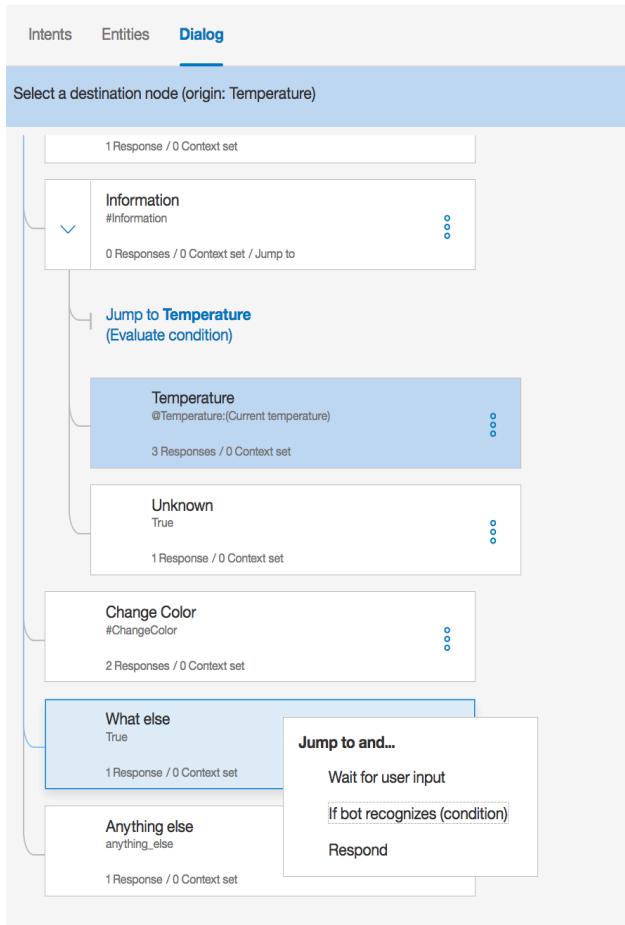


47. Next we want to select the **3 blue buttons** on the **Temperature** node. Like with the "Information" node, select the **Jump to** menu item.

Workshop: Watson Conversation Service

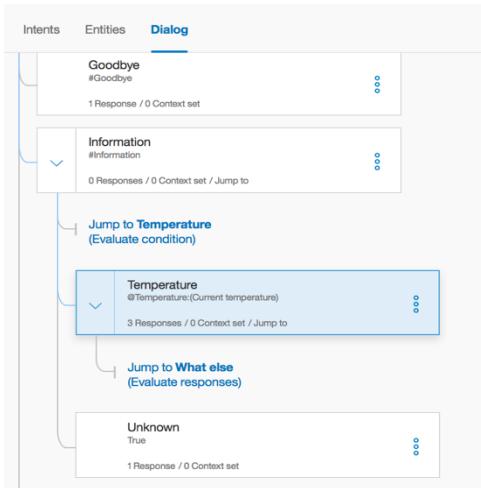


48. Then select the **What else** node. This time select the **Respond** option.



49. The sub-flow should now look like the following:

Workshop: Watson Conversation Service



50. We are now finished with the entire Dialog. Congratulations! Now is time to test your code.

Weather Company Service & NodeRed Flow

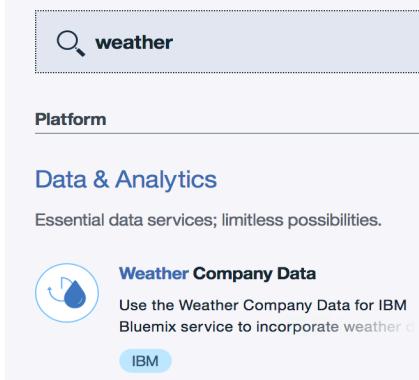
What might become obvious is that the responses from WCS are static. There is nothing personalized in the responses. Because of this we need to add a layer in front of WCS to take the responses and make them personalized. For this, we are going to use NodeRed. We are also going to use the Weather Company Service to determine the weather at our location (Grapevine, TX).

Create Weather Service

1. Navigate back to your **bluemix dashboard** tab and select **Catalog** from the upper right corner



2. In the search, type **weather** and select the **Weather Company Data** service



Workshop: Watson Conversation Service

3. In the service name type **WK1-Weather-xx** (where xx is your team number). You can leave credential name as is, if you like. Click on **Create** in the lower right corner.

Weather Company Data

This service lets you integrate weather data from The Weather Company into your IBM Bluemix application. You can retrieve weather data for an area selected by a geolocation. The data allows you to build applications that solve real business problems and have a significant impact on the bottom line.

WARNING: Currently, the Weather Company Data for IBM Bluemix service is MYT NOT purchased or used in the following countries or regions: Afghanistan, Armenia, Azerbaijan, Bahrain, Bangladesh, Bhutan, Brunei, Cambodia, China, Cyprus, Georgia, Indonesia, Iran, Iraq, Japan, Jordan, Kazakhstan, Kuwait, Kyrgyzstan, Laos, Lebanon, Malaysia, Maldives, Mongolia, Myanmar, Nepal, Oman, Pakistan, Qatar, Russia/Saudi Arab, Singapore, South Korea, Sri Lanka, Syria, Taiwan, Tajikistan, Timor-Leste, Turkmenistan, United Arab Emirates, Uzbekistan, Vietnam, Yemen. We encourage you to check back often to review the list. We will update it when additional information becomes available.

Service name: <input type="text" value="WWT-Weather-01"/>	Credential name: <input type="text" value="Credentials-1"/>	Select region to deploy in: <div style="border: 1px solid #ccc; padding: 2px; width: 100%;"><input type="text" value="US South"/> ▼</div>
Choose an organization: <input type="text" value="lmurphy@us.ibm.com"/>		Choose a space: <div style="border: 1px solid #ccc; padding: 2px; width: 100%;"><input type="text" value="Workshop"/> ▼</div>
Connect to: <div style="border: 1px solid #ccc; padding: 2px; width: 100%;"><input type="text" value="Leave unbound"/> ▼</div>		

4. You should now see the following

Data & Analytics / WK1-Weather-01



Weather Company Data

Use Weather Company Data for IBM Bluemix to integrate weather data from The Weather Company into your IBM Bluemix applications. This service lets you retrieve weather data for an area specified by a geolocation. The data allows you to create applications that solve real business problems where weather has a significant impact on the outcome.

5. You have now completed creating the Weather Company Data service instance.

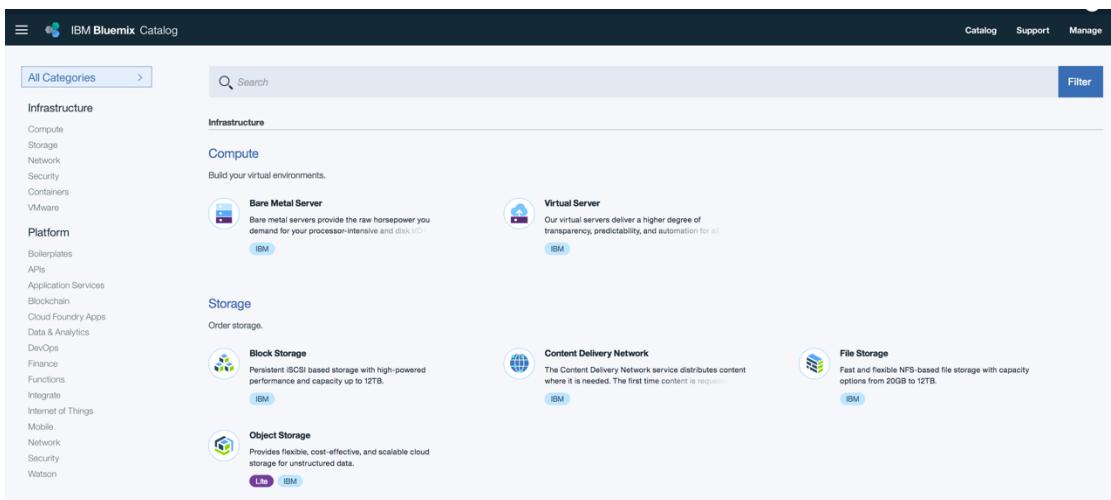
Create NodeRed Flow

Next, we are going to create a **Bluemix NodeRed** application.

1. Navigate back to your **bluemix dashboard** tab, and select **Catalog** from the upper right corner.

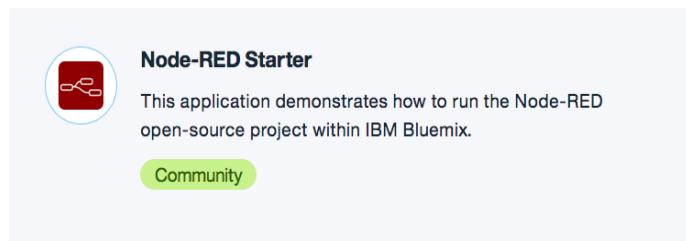


2. You should now see this



Workshop: Watson Conversation Service

3. Select **Boilerplates** on the left navigation under the Platform menu and select **Node-RED Starter**



4. On the resulting page, you need to give your application a name. You can name your application anything you like provided it is unique in the Bluemix space. For consistency, this workshop will use the name **myWorkshop-xx** (Replace xx with your team number). Enter your application name in the application name field, and click **Create**. There is no need to modify any of the other fields on this page.

A screenshot of the Node-RED Starter creation page in the IBM Bluemix Catalog. The page has a dark header with "IBM Bluemix Catalog" and "Catalog Support Manage" buttons. The main form includes fields for "App name" (set to "myWorkshop-01"), "Host name" (set to "myWorkshop-01"), "Domain" (set to "mybluemix.net"), "Select region to deploy in" (set to "US South"), "Choose an organization" (set to "lrmurphy@us.ibm.com"), and "Choose a space" (set to "Group Up Workshop"). Under "Selected Plan", "SDK for Node.js™" is chosen for the "Default" plan, and "Cloudant NoSQL DB" is chosen for the "Lite" plan. At the bottom, there are links for "Need Help?", "Contact Bluemix Sales", "Estimate Monthly Cost", and "Cost Calculator", along with a large blue "Create" button.

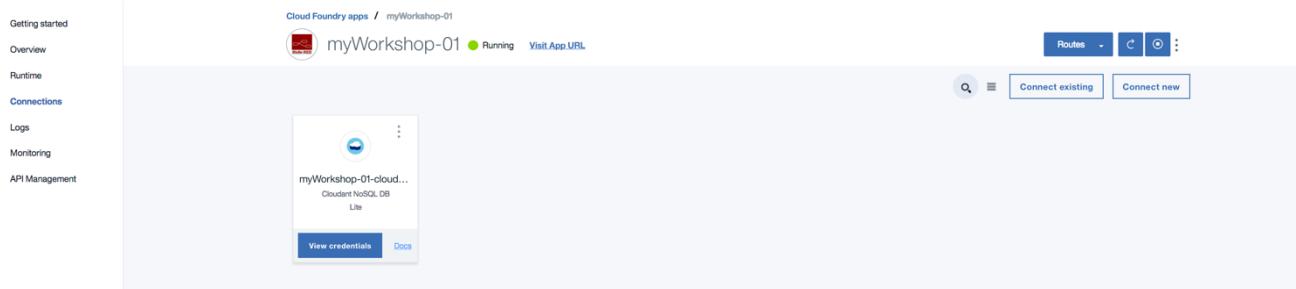
5. At this point, your application will be created and, after a few moments, you will be taken to the **Getting Started** page for your application.

A screenshot of the "Getting started" page for the "myWorkshop-01" application. The left sidebar lists "Getting started", "Overview", "Runtime", "Connections", "Logs", "Monitoring", and "API Management". The main content area shows the application's status as "Running" with a green dot. It includes a "Start coding with Node-RED" section with a "Last Updated: 2017-06-15" message. Two numbered steps are provided: ① After your application has started, click on the **Routes URL** or enter the following URL in a browser: `http://<yourhost>.mybluemix.net` (with a copy icon) and ② Click **Go to your Node-RED flow editor**. This opens up a browser-based flow editor that makes it easy to wire together devices, APIs, and online services by using the wide range of nodes included in its palette.

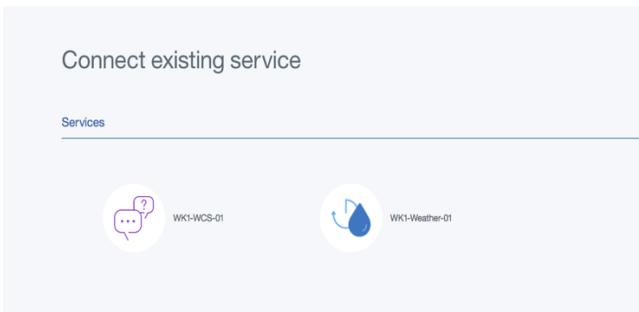
6. You now need to create connections to your Weather Service and Watson Conversation services. Go to the **Connections** section of your application page (listed on the menu to the left). There, as

Workshop: Watson Conversation Service

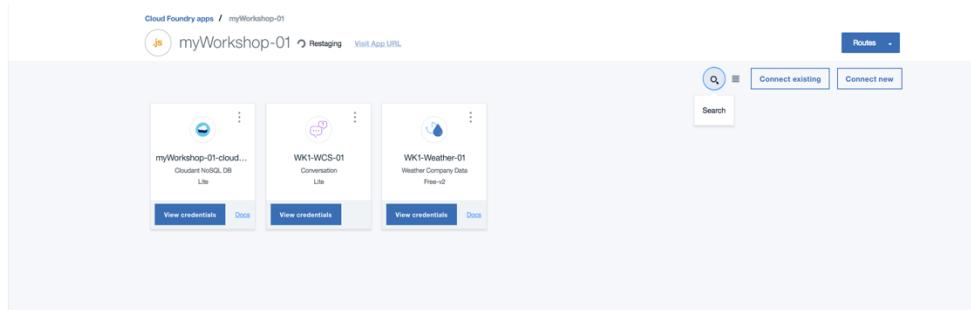
a part of the Node-RED Starter boilerplate, you will see that a Cloudant NoSQL database service has been created automatically. The Cloudant NoSQL database holds all of your application code. Here, is where you will add the connections to your two additional services.



7. Select **Connect Existing**, on the right, and connect both your Weather and Conversation service. Select **Restage** if prompted to restage the application.

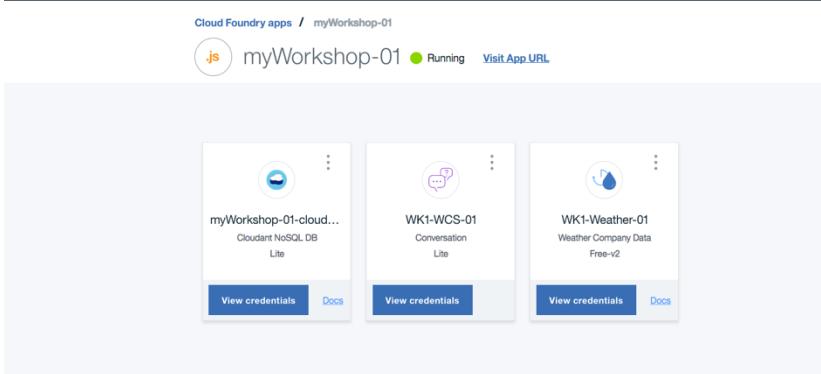


8. You should now having the following:



9. Next, once your application is running, select **Visit App URL** to open the NodeRed editor.

Workshop: Watson Conversation Service



10. Follow the "welcome to NodeRed steps" to secure your Node-Red editor. **Write down your username and password, as they will be needed to log into the Editor.**

Welcome to your new Node-RED instance on IBM Bluemix

We know you're eager to start wiring up your flows, but first there are a couple of tasks you should do:

- Secure your Node-RED editor
- Browse available IBM Bluemix nodes

11. Select **Finish** to complete the install

Finish the install

You have made the following selections:

- Secure your editor so only authorised users can access it

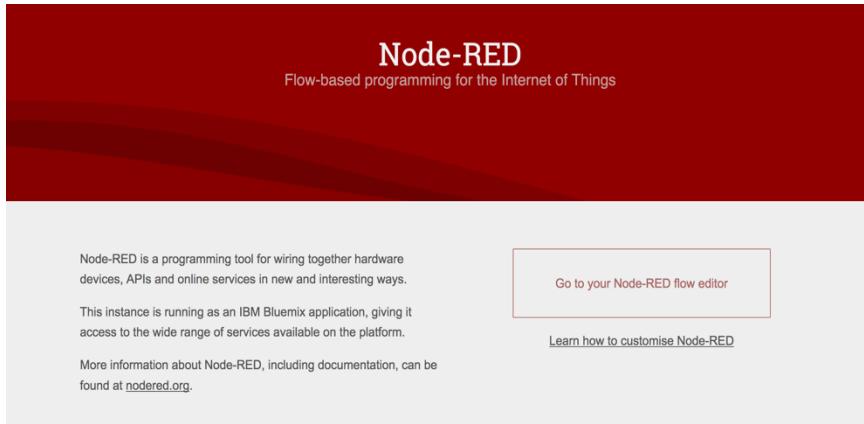
You can change these settings at any time by setting the following environment variables via the Bluemix console:

- NODE_RED_USERNAME - the username
- NODE_RED_PASSWORD - the password
- NODE_RED_GUEST_ACCESS - if set to 'true', allows anyone read-only access to the editor

Previous **Finish**

12. Upon a successful install, you will be taken to your NodeRed Editor homepage.

Workshop: Watson Conversation Service

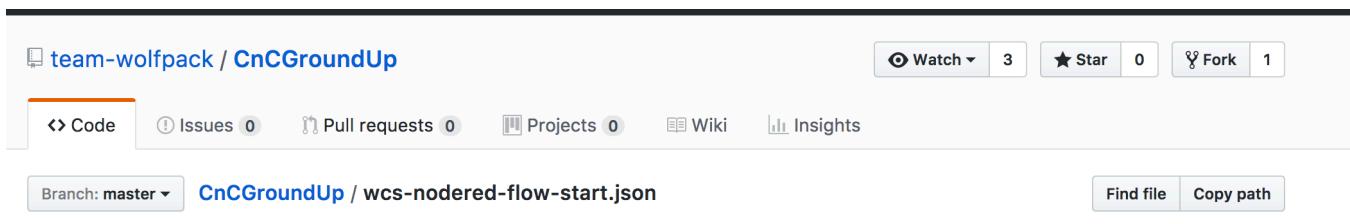


The screenshot shows the Node-RED landing page. At the top, it says "Node-RED" and "Flow-based programming for the Internet of Things". Below that, there's a message: "Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways." It also states: "This instance is running as an IBM Bluemix application, giving it access to the wide range of services available on the platform." A link "Go to your Node-RED flow editor" is present, along with a link "Learn how to customise Node-RED".

13. Select **Go to your Node-RED flow editor** and enter the username and password that was previously created. You should now see the following:



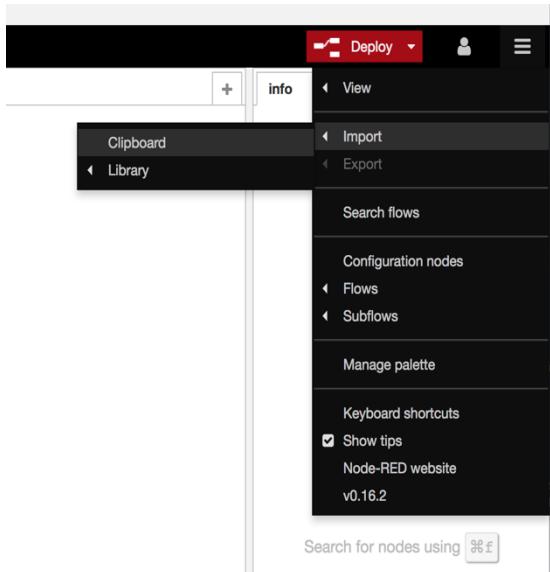
14. Click [WCS NodeRed Flow](#) link. This will take you to the NodeRed flow JSON file which has a pre-developed flow for talking to WCS. You should now see:



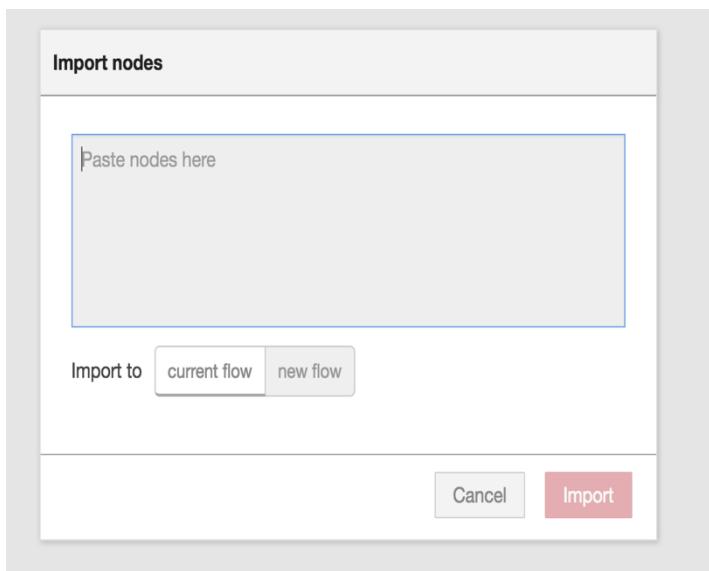
The screenshot shows a GitHub repository page for "team-wolfpack / CnCGroundUp". The repository has 3 stars, 0 forks, and 1 issue. The "Code" tab is selected. The URL is "Branch: master / CnCGroundUp / wcs-nodered-flow-start.json". There are buttons for "Find file" and "Copy path".

15. Highlight all of the JSON text and copy it to your clipboard. (ctrl+c on your keyboard).
16. Go back to your NodeRed editor and in the upper right corner is the cake layer icon, click on it. Then click **import** and then click **clipboard**

Workshop: Watson Conversation Service

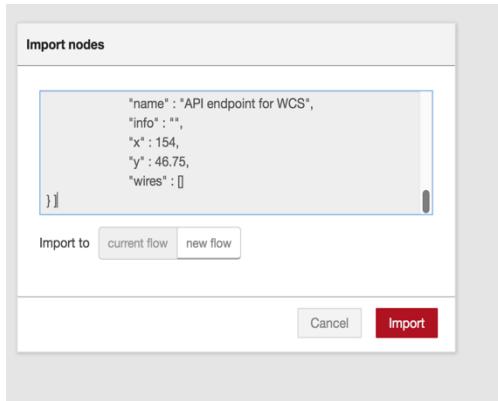


17. You should now see the following dialog window:

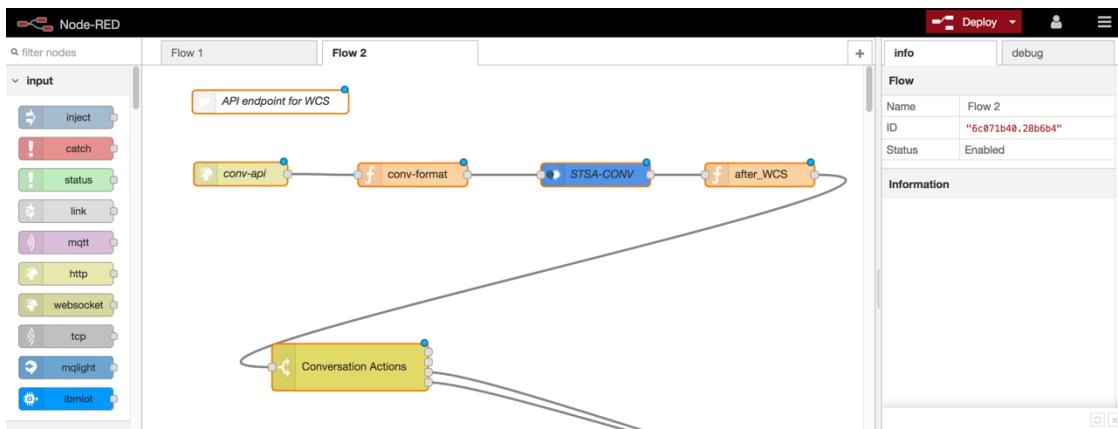


18. Click in the center of the dialog and paste the information from your clipboard. (ctrl+v). Make sure the **new flow** button is pressed before you import.

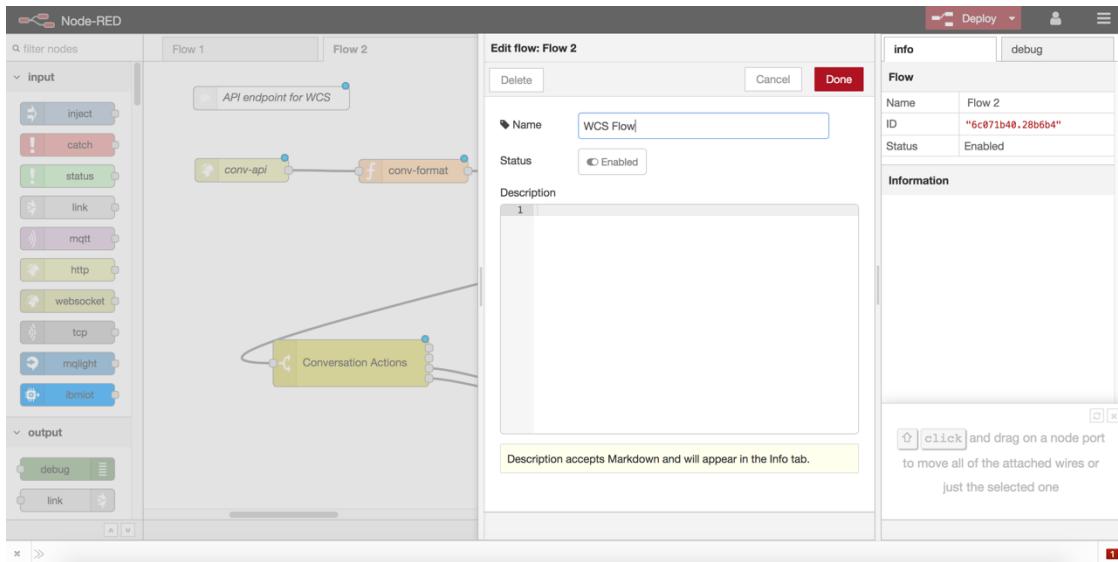
Workshop: Watson Conversation Service



19. Your screen should now look something like the following



20. Double click the **Flow 2** tab, to edit the name of the Flow. Change the name to **WCS Flow** and select **Done**.



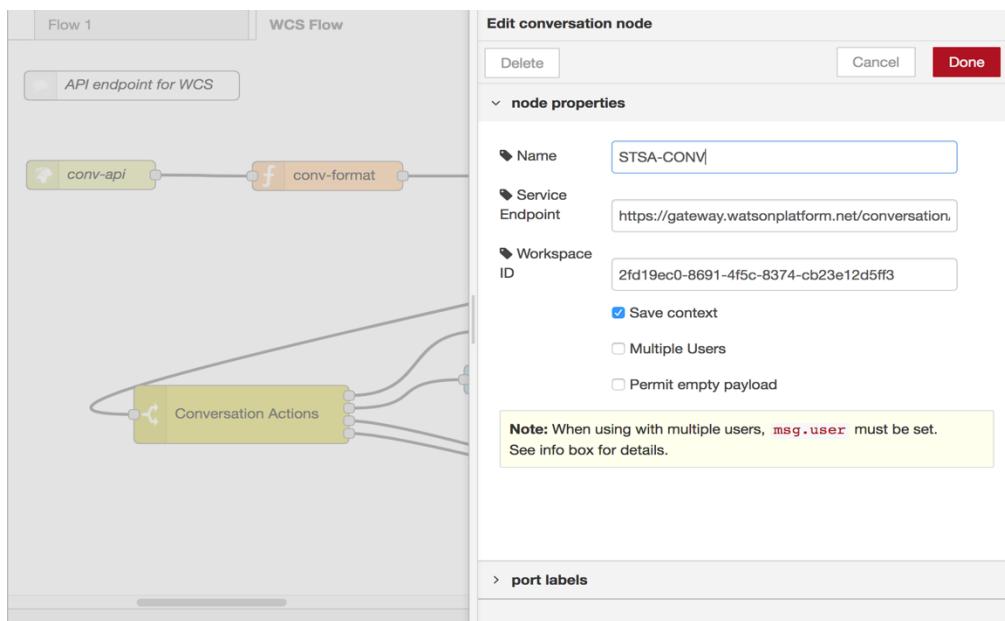
21. Congratulations, you have created a NodeRED service and imported a starter flow.

Workshop: Watson Conversation Service

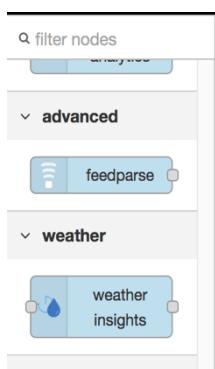
Update NodeRed Flow

Now we need to update a couple of the nodes to include the Watson Conversation and Weather Company service.

1. Now go back to NodeRed and make sure you are in the **WCS Flow** tab.
2. First we will update the Watson Conversation node to point to the conversation service we previously created. Double click the **STSA-CONV** node. Replace the information in the **Workspace ID** with the workspace ID from your conversation service. You recorded this value earlier in the workshop. Click **Done**

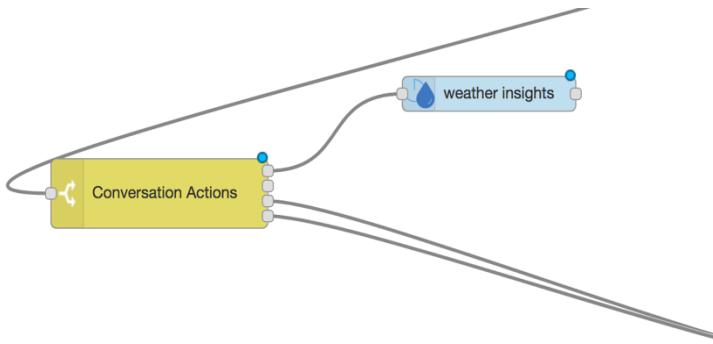


3. On the left-side navigation panel, under **Weather**, select the **Weather Insights** node.



4. Drag and drop the Weather Insights node onto the workspace. Connect the **Weather Insights** node to the **Conversation Action** node by clicking and dragging from one connection point to the other.

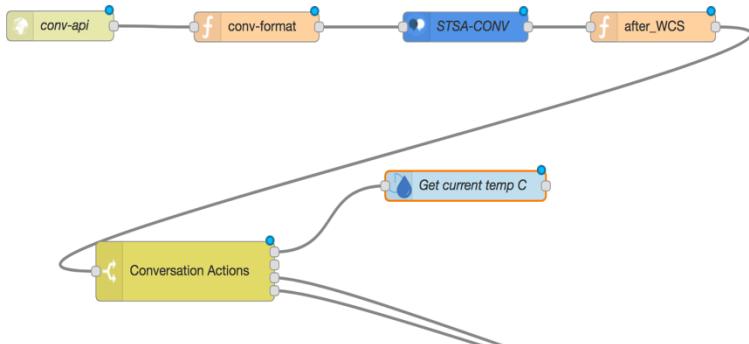
Workshop: Watson Conversation Service



- Double click the **Weather Insights** node, to edit the node properties to the following. For purposes of this lab, we are hardcoding the geo-coordinates (32.9343,-97.0781) of Grapevine, TX. The weather service will return the weather for that location. Select **Done** when finished.

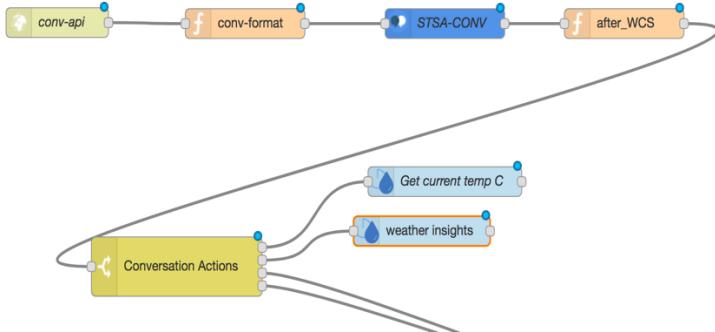
Name	Get current temp C
Service	Current Observations
Location	32.9343,-97.0781
Units	Metric
Language	en-US

- Notice the name of the Weather Insights node on the workspace, is now **Get current temp C**



- We now need to add another weather Insights node to get weather in Fahrenheit. Drag and drop another **Weather Insights** node onto the workspace and connect it to the Conversation Action node.

Workshop: Watson Conversation Service



8. Double click the node to edit the properties to the following. Select **Done**

Flow 1 WCS Flow Edit weather insights node

API endpoint for WCS

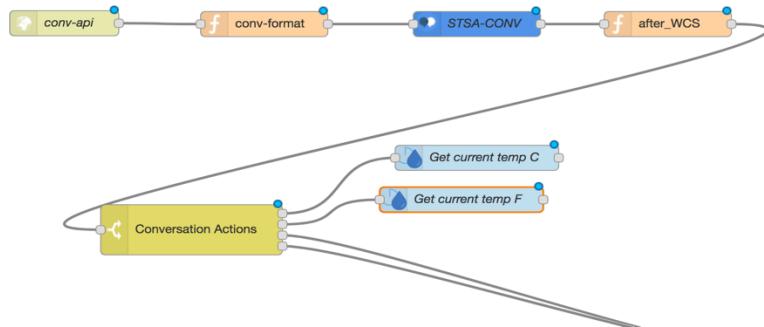
conv-api --> conv-format

Conversation Actions

Name: Get current temp F
Service: Current Observations
Location: 32.9343,-97.0781
Units: Imperial (English)
Language: en-US

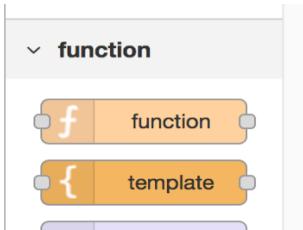
port labels

9. Again, notice the name of the node is now **Get current temp F**

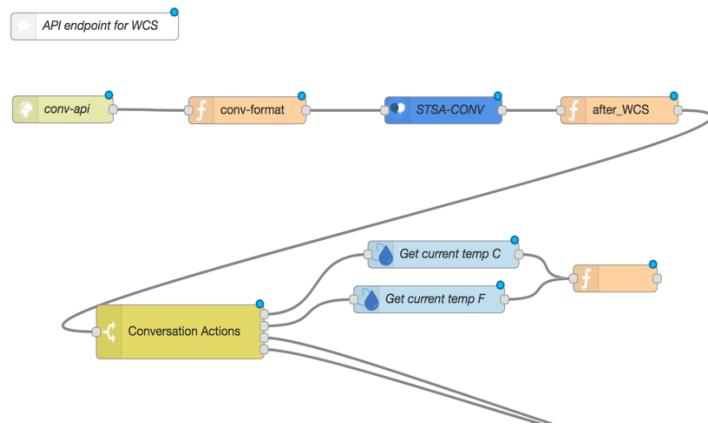


10. Next, we need to format the information returned from the Weather Service. For this, we will use a **Function** node. On the left-side of the workspace under "Function", select the **function** node.

Workshop: Watson Conversation Service



11. Drag and drop the function node onto the workspace and connect to it the **Get current temp C** and **Get current temp F** nodes. You should now have the following:



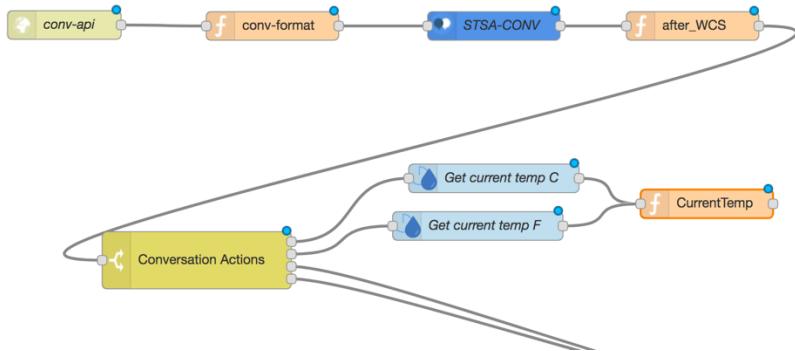
12. Double click the function node to edit properties. Name the node **CurrentTemp** and input the following code into the function box. Select **Done** when finished.

```
txt = msg.convdata.output.text[0];
temp = msg.observation.temp;
msg.payload = msg.convdata;
msg.payload.output.text[0] =  txt + " " + temp;
return msg;
```

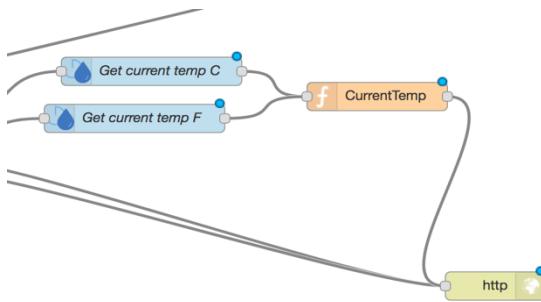
The screenshot shows the workspace with the "Edit function node" dialog open over the workspace. The dialog has the following fields:
- **Name:** CurrentTemp
- **Function:** (contains the provided JavaScript code)
- **Outputs:** 1
- **Port Labels:** (empty)

13. Your workspace should now look like the following:

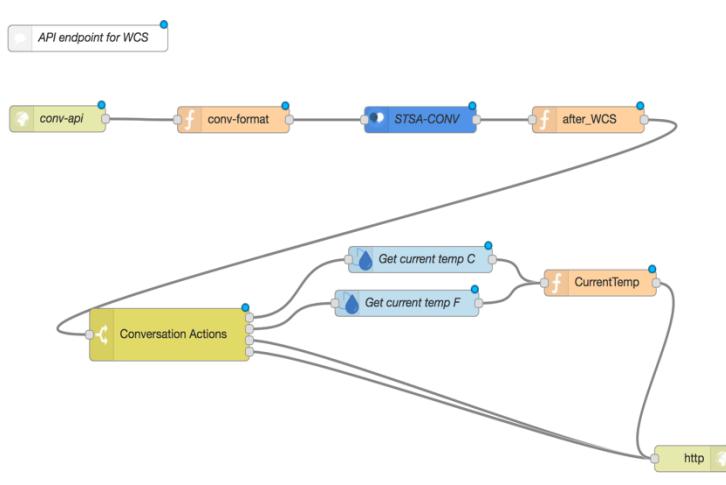
Workshop: Watson Conversation Service



14. The final step is to connect the **CurrentTemp** node to the **http** node



15. Your final NodeRED flow should look like:



16. You are done updating the NodeRed Flows for Watson Conversation Service and the Weather Service. Click the **DEPLOY** button at the top of the screen. You should receive a "Successfully deployed" message.

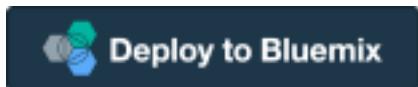
Workshop: Watson Conversation Service



Watson Conversation Web Client

The next step is to create a sample application that connects to your NodeRed flow. When you click on the button below, you will be taken to bluemix DevOps page. This will automatically create a new application for you, via the DevOps ToolChain. If you are not logged on to BlueMix you will need to logon.

1. Click the Deploy to BlueMix button below to provision the web client application.



2. You will see a screen like the following:

After you click **Deploy**, your app will be deployed to Bluemix.
Your app's code will be automatically loaded into a Git repo. Each time you commit changes to the repo, they are automatically deployed by using a toolchain that is associated with your app. You can add more tools to the toolchain and share it with your team. [Learn more](#).

The toolchain is part of the Continuous Delivery service. If an instance of this service can't already run in your organization, when you click **Deploy**, it is automatically added at no cost to you. For more information and terms, see the [Bluemix catalog](#).

Region	Organization	Space
US South (Production)	jdcalus@us.ibm.com	dev

3. Click the **deploy** button, after about 1 minute or so, but may be longer if BlueMix is slow :-), your screen will change to something like the following:

Workshop: Watson Conversation Service

STSA-WCS_WebProxy-20170609172152769

Your app is being created! Quick start: To watch the pipeline deploy your app, click [Delivery Pipeline](#). After the app is deployed, you can see it running by clicking [View app](#).

View app

4. Click on the **Deliver** tile

DELIVER

Delivery Pipeline
STSA-WCS-WebProxy...

✓ Configured

5. Then you should see something like the following:

STSA-WCS-WebProxy-20170609173417132 | Delivery Pipeline

Build Stage

STAGE PASSED

LAST INPUT Git URL
Last commit by jcalus
feed underscore problem

JOB Build Passed now

LAST EXECUTION RESULT Build 1

Deploy Stage

STAGE RUNNING... [1 QUEUED]

LAST INPUT Stage: Build Stage / Job: Build

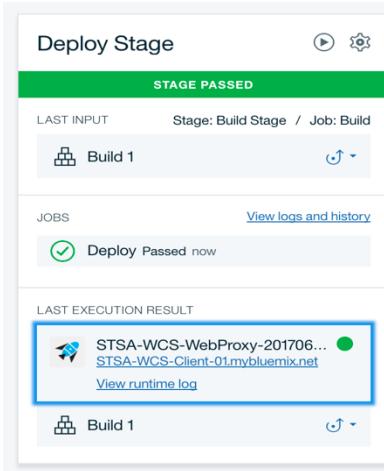
JOB Deploy Running

LAST EXECUTION RESULT No results

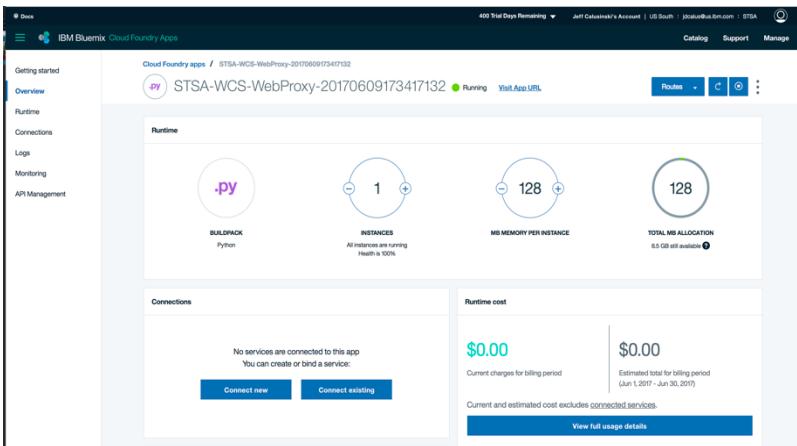
Add Stage

6. Once the message in the deploy tile has changed to Passed now from Deploy running, click on the **Rocket** icon under **Last Executed Results**

Workshop: Watson Conversation Service



7. This will take you to the newly created applications. Your screen should now look like:



8. Click on the **Runtime** link on the left and your screen should change, then click on **environment variables** tab in the middle of the screen. It should look like the following:

Workshop: Watson Conversation Service

The screenshot shows the IBM Bluemix Cloud Foundry Apps interface. On the left, a sidebar lists 'Getting started', 'Overview', 'Runtime' (which is selected), 'Connections', 'Logs', 'Monitoring', and 'API Management'. The main area displays the 'Cloud Foundry apps / STSA-WCS-WebProxy-20170609173417132' page. It indicates the app is 'Running' and provides a 'Visit App URL'. Below this, there are tabs for 'Memory and instances', 'Environment variables' (which is selected), and 'SSH'. Under 'VCAP_SERVICES', it says 'The value for VCAP_SERVICES is empty. Either there are no services associated with this app or you are not authorized to view them.' There is an 'Expert' button. Under 'User defined', there is a table:

NAME	VALUE	ACTION
CONVERSATION_PASSWORD	value3	X
CONVERSATION_URL	https://xxxxxxxxxx.mybluemix.net/v1/workspaces/voiceproxy/message	X
CONVERSATION_USERNAME	value2	X

At the bottom are 'Add', 'Save', 'Reset', and 'Export' buttons.

9. In the **CONVERSATION_URL** value change the "xxxxxx" to the hostname of your **NodeRed** application. This comes from the NodeRed url where you created your NodeRed flow. So something like "myworkshop-xx.mybluemix.net", do not include "/red/#". Change the **CONVERSATION_PASSWORD** and **CONVERSATION_USERNAME** value to the password and username from your Watson Conversation Service. You recorded these values earlier in the lab.

This is a screenshot of the 'User defined' environment variables table from the previous screenshot. It shows the same three variables: CONVERSATION_PASSWORD, CONVERSATION_URL, and CONVERSATION_USERNAME, each with its corresponding value and an 'X' icon in the ACTION column.

NAME	VALUE	ACTION
CONVERSATION_PASSWORD	RgcrJoOMuk8Z	X
CONVERSATION_URL	https://myworkshop-01.mybluemix.net/v1/workspaces/voiceproxy/message	X
CONVERSATION_USERNAME	0c54cd07-6d0a-45b1-bc59-8f959a66e646	X

Your values will be different

10. Click the **Save** button. The application will restart.
11. Once your application is running, click on the **Visit App URL**. This is at the top of the page. **You will get an error message** or land on the following page.

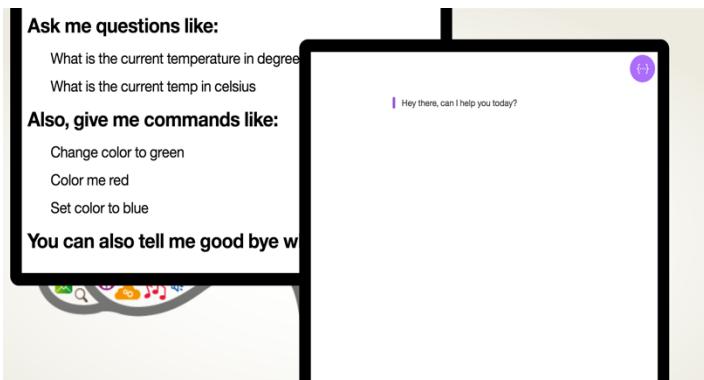
This is a screenshot of a web browser window. The address bar shows a secure connection to 'https://stsa-wcs-webproxy-20170705214816338-floaty-tohubohu.mybluemix.net'. The page content is a standard 'Not Found' error message: 'Not Found'. Below the address bar, there are links for 'Apps', 'Bookmarks', and 'Mail'.

Not Found

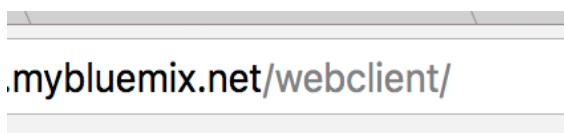
The requested URL was not found on the server. If you entered the URL manually please check your spelling and try again.

or

Workshop: Watson Conversation Service



12. That is okay. You need to add **webclient** to the end of the browsers URL.



13. Your screen should now look like the following:

```
Watson understands
1 {
2   "intents": [
3     {
4       "confidence": 1,
5       "intent": "Greeting"
6     }
7   ],
8   "entities": {},
9   "context": {
10     "conversation_id": "b844fb8b-3ba6-4a94-be45-b033b3fc0d73",
11     "system": {
12       "dialog_stack": [
13         {
14           "dialog_node": "root"
15         }
16       ],
17       "dialog_request_counter": 9,
18       "dialog_turn_counter": 9,
19       "branch_exited": true,
20       "node_output_map": {
21         "node_1_1495449712882": {
22           "text": "Hello, how can I help you today?"
23         },
24         "node_1_1495396735644": {
25           "text": "I'm sorry, I didn't understand that. Could you please rephrase your question or provide more context?"
26         },
27         "node_2_1495584219351": {
28           "text": "Is there anything specific you would like to know or discuss? I'm here to help with various topics such as weather, news, or general information."
29         },
30         "node_12_1495597223065": {
31           "text": "I'm sorry, I didn't understand that. Could you please rephrase your question or provide more context?"
32         },
33         "node_1_149539499373": {
34           "text": "I'm sorry, I didn't understand that. Could you please rephrase your question or provide more context?"
35         },
36         "node_2_1495443824562": {
37           "text": "Is there anything specific you would like to know or discuss? I'm here to help with various topics such as weather, news, or general information."
38         },
39       },
40       "branch_exited_reason": "completed"
41     }
42   },
43   "input": {
44     "text": "Hello"
45   },
46   "output": {
47     "log_messages": [],
48     "text": [
49       "Hey there, how can I help you today?"
50     ],
51     "node_visited": []
52   }
53 }
```

Ask Watson questions

You can now ask Watson questions about the temperature.

1. What is the current temperature in Celsius
2. What is the current temperature
3. What is the current temperature in fahrenheit
4. Change the background color to blue

Workshop: Watson Conversation Service

Ask Watson this question

5. What is the temperature

Did you get the response you were expecting? Why not? (hint: go look at the Temperature dialog node in Watson Conversation)

Nice Job you are now a Jedi Knight in Watson Conversation Service!

Jedi Knights and Jedi Masters

If you have time, you can start to add more Dialog Nodes to ask other questions and you can provide the appropriate responses. Play with the ordering of the nodes in the dialog tree. Create new child nodes and see how they react to the flow. Also you can always add new entities and intents and then create even more dialog nodes. It is really endless as to what you can do.